Actility
Connecting with intelligence

Actility ThingPark Enterprise
LRC-AS Tunnel Interface Developer Guide (LoRaWAN®)

Under NDA

NOTICE

This document contains proprietary and confidential material of ACTILITY SA. This document is provided under and governed by either a license or confidentiality agreement. Any unauthorized reproduction, use, or disclosure of this material, or any part thereof, is strictly prohibited.

The material provided in this document is believed to be accurate and reliable. However, no responsibility is assumed by Actility SA for the use of this material. Actility SA reserves the right to make changes to the material at any time and without notice. This document is intended for information and operational purposes only. No part of this document shall constitute any contractual commitment by Actility SA.

Portions of this documentation and of the software herein described are used by permission of their copyright owners.

Actility, ThingPark, are registered trademarks of Actility SA or its subsidiaries may also be registered in other countries.

Other denoted product names of Actility SA or other companies may be trademarks or registered trademarks of Actility SA or its subsidiaries, or their respective owners.

Headquarters

Actility Lannion,

Actility S.A 4 rue Ampère BP 30225

22300 Lannion France

www.actility.com

VERSIONS

Version	Date	Author	Details
01	09/07/2015	Actility	Initial Version
02	12/11/2015	Actility	Update with https
03	27/11/2015	Actility	Connectivity
04	04/01/2016	Actility	Update for Downlink security.
06	21/01/2016	Actility	Application Server authentication example clarification.
07	01/02/2016	Actility	Operator domain name clarification.
08	01/02/2016	Actility	Update SHA-256 token in the example.
09	22/03/2016	Actility	Reordering within the downlink security token to fix the Hash length extension vulnerability.
10	04/04/2016	Actility	Rewriting of section 2 (Connectivity).
11	21/06/2016	Actility	Fixing errors in the authentication examples.
12	01/12/2016	Actility L. Guillemot	In the URL of the HTTP request, use ASCII codes in the Time parameter when the uplink/downlink security is enabled. See *5.1 ISO 8601 Timestamps*.
13	22/12/2016	Actility L. Guillemot	Clarification about LRC-AS key entropy *in 4 Connectivity*. <DevAddr>, <AckRequested>, <ADRAckReq>, <FPending>, and <rawMacCommands> always returned in the Uplink frame report and the Downlink frame sent report. For more information, see *6.1.3 Document Fields* and *6.2.3 Document Fields*.
14	27/01/17	Actility L. Guillemot	The document title is simplified. When the uplink/downlink security is activated, the timestamp parameter is encoded differently in SHA256 and URL. See *5.1.2 Timestamp Encoding When Uplink/Downlink Security Is Activated*.
15	12/09/17	Actility P.Paysant-Le Roux	Tunneling interface enhancements for **ThingPark 4.3**. See *8.1.6 What's New in Release 4.3*.
16	13/10/17	Actility ML Ancelle	LrnDevEui value should be written in lower case. See *6.5.2 Token Verification Details*.
17	16/01/18	Actility S. Dufour	Tunneling interface enhancements for **ThingPark 5.0**. See *8.1.5 What's New in Release 5.0*.

Version	Date	Author	Details
18	08/03/18	Actility S. Dufour	Fix Uplink frame report XML Elements: ■ Rename <DevNorthVelocity> as <DevNorthVel> ■ Rename <DevEastVelocity> as <DevEastVel>
19	29/03/2018	Actility S. Dufour	Renamed document as "LRC-AS Tunnel Interface Developer Guide (LoRaWAN®)"
20	20/04/2018	Actility R. Soss S. Dufour	Tunneling interface enhancements for **ThingPark 5.1**. See *8.1.4 What's New in Release 5.1* ■ Syntax of CustomerData attribute in Uplink Frame Report and Downlink Frame Sent Report has been clarified. ■ Added important notes regarding HTTPS session handling.
21	04/06/2018	Actility S. Dufour	■ Fix Uplink frame report attributes: ○ Add <DevAltDilution>
22	12/09/2018	Actility M. Denalie	■ Renamed the document as "TP_Enterprise_5.1-rev1_LRC-AS Tunnel Interface Developer Guide_LoRaWAN"; ■ Delete the following sections: ○ "Uplink Kafka/ Downlink HTTP" connectivity option ○ LRC to AS: Multicast Summary Report + Location Report ○ AS to LRC: Downlink Multicast ○ - Appendix – Effect of SSM/HSM Modes on AS Development
23	25/01/2019	Actility S. Dufour	Tunneling interface enhancements for **ThingPark 5.2.2**. See *8.1.3 What's New in Release 5.2.2*. Added missing Lrrs/Lrr/Chain field in uplink frame report. See *6.1.3 Document Fields*. Fixed sample downlink CURL command to include AS authentication..
24	27/06/2019	Actility S. Dufour	Added syntax indication for all document fields on LRC to Application Server Tunnel interface. Clarified the Margin encoding rule in Uplink Frame Report.

Version	Date	Author	Details
25	01/07/2019	Actility S. Dufour	Tunneling interface enhancements for **ThingPark 6.0**. Added syntax indication for all query parameters. Fixed the FPort range to `1..224`.
26	09/08/2019	Actility S. Dufour	Added new response codes: ■ `350 "Invalid payload. Must not be empty"` ■ `350 "Payload must be provided encrypted with the downlink counter value"`
27	20/09/2019	Actility S. Dufour	Clarified the reporting of the `FCntDn` field and the `AFCntDn` field in Uplink Frame Report. Fixed Tunnel Interface Authentication Key examples.
28	11/12/2019	Actility S. Dufour	Tunneling interface enhancements for **ThingPark 6.1**. See *8.1.1 What's New in Release 6.1*.
29	04/02/2020	Actility S. Dufour	Clarified in which conditions an uplink packet is reported to the Application Server. Fixed the FPort range to `0..224` for uplink frame report. Added token verification details for location report and notification report.
30	14/09/2020	Actility S. Dufour	Notification report: clarified the content of `Time` attribute and the conditions to trigger the report.
31	20/11/2020	Actility S. Dufour	RDTP-15549: the downlink frame interface can be used to only purge the AS downlink queue of the device; in this case no payload is provided. RDTP-6289: new downlink frame error code for token bucket limit reached.
32	09/12/2020	Actility S. Dufour	"What's New in Release 6.1" section updated according to RDTP-15549 and RDTP-6289

REFERENCE DOCUMENTS

	Documents	Author
01	LoRaWAN® Specification	*LoRa Alliance®*
02	ThingPark Wireless Device Manager User Guide	Actility - L. Guillemot
03	ThingPark Wireless Network Manager User Guide	Actility
04	ThingPark Wireless Operator User Guide	Actility - L. Guillemot
05	ThingPark Wireless Supplier User Guide	Actility
06	ThingPark Wireless Vendor User Guide	Actility

TABLE OF CONTENTS

DEFINITIONS AND ACRONYMS

Acronyms	Definitions
ABP	Activation-By-Personalization
ACK	Acknowledgement of an alarm
ADR	Automatic Data Rate
AES	Advanced Encryption Standard
AS	Application Server
BPM	Business Process Management
BSS	Billing Support Systems
CSP	Communication Service Provider
End Device	A sensor or actuator
ESP	Estimated Signal Power
ETSI	European Telecommunications Standards Institute
FCtrl	Frame Control
GSCL	Gateway Service Capability Layer
GTM	Go To Market
HAN	Home Area Network
HSM	Hardware Security Module
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure As A Service
IEC	International Electrotechnical Commission
IoT	Internet of Things
ISM	Industrial Scientific Medical
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LAT	Latitude
LC	Logical Channel
LON	Longitude
LoRaWAN®	Long Range Wide Area Network
LPWAN	Low Power Wide Area Network

Acronyms	Definitions
LRC	Long Range Controller
LRR	Long Range Relay
M2M	Machine-to-Machine
MAC	Media Access Control
MIC	Message Integrity Code
MTBF	Mean Time Before Failure
NAT	Network Address Translation
NW	Network
OBIX	Open Building Information Exchange
OSS	Operations Support Systems
OTA	Over The Air
OTAA	Over-The-Air-Activation
PER	Packet Error Rate
PKI	Public Key Infrastructure
POC	Proof Of Concept
REST	Representational State Transfer
RF	Radio Frequency
RIT	Receiver Initiated Transmit
RSSI	Received Signal Strength Indicator
Rx	Receiver
RX1	First receive window
RX2	Second receive window
SaaS	Software As A Service
SF	Spreading Factor
SLRC	Secured LRC (VPN Concentrator)
SMP	System Management Platform
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SNR	Signal to Noise Ratio
SSO	Single Sign On

Acronyms	Definitions
TLS	Transport Layer Security
TWA	ThingPark Wireless Application
Tx	Transmitter
UNB	Ultra Narrow Band
URL	Uniform Resource Locator
VM	Virtual Machine
VPN	Virtual Private Network
XML	Extensible Markup Language

CONVENTIONS

This document uses the following conventions.

Convention	Description
Domain name	For readability, the `api.thingpark.com` domain name is used to illustrate examples in this document. However, according to the deployed platform, this domain name must be understood as the operator domain name (e.g. `api.my-operator-domain.com`).

1 SCOPE

The scope of this document is to give development guidelines on the tunnel mode interface to applications developers. This document only targets LoRaWAN® access network.

This document is fully applicable to ThingPark Enterprise OCP and SaaS Editions for Generic Applications.

2 THINGPARK SOLUTION OVERVIEW

2.1 ThingPark Solution Architecture Description

The ThingPark set consists of four main key components:

- ThingPark Wireless – Core network and OSS
- ThingPark OS
- ThingPark X
- ThingPark Enterprise

The ThingPark platform is a **modular solution** enabling Network Operators to:

- Deploy LPWANs based on LoRaWAN® or LTE with **ThingPark Wireless**.
- Manage, activate and monetize IoT bundles (device, connectivity and application) with **ThingPark OS**.
- Provide value-added data layer services, such as protocol drivers and storage with **ThingPark X**.

ThingPark OSS acts as the central System Management Platform (SMP), enabling all other ThingPark platform modules with base capabilities such as subscriber management, centralized authentication and access rights, and workflow management.

ThingPark Enterprise is an Internet of Things (IoT) platform that manages private LoRaWAN® Networks. The ThingPark Enterprise edition is used by companies to support their specific business.

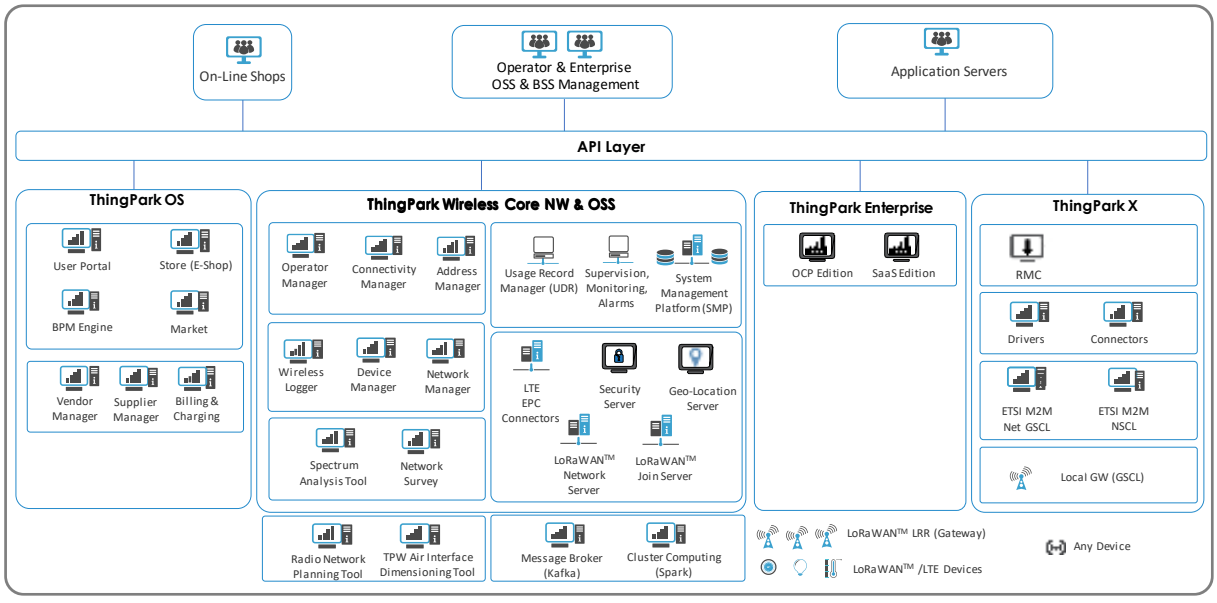


Figure 1 - ThingPark Solution Architecture Description High Level Product Illustration
Note that the modules above may be representing a physical server, a function, a service or a business support layer as part of the overall ThingPark solution and not necessarily a physical HW server.

3 THINGPARK ENTERPRISE TUNNEL MODE INTERFACE

The following interface is provided by the ThingPark Enterprise platform for developers of applications interfaced with wireless sensors or actuators compatible with the LoRaWAN® specification.

- **Tunnel mode interface**: It is a simple message-passing interface between the ThingPark Enterprise servers which implement the network MAC layer (LRC servers) and Application Servers. This interface forwards the uplink radio packets raw payload data and associated metadata (RSSI, SNR…) to one or more Application Servers associated to the end device IEEE EUI (DevEUI). As ThingPark Enterprise supports bidirectional communication, Application Servers may also send requests to one of the LRC nodes to send downlink frames to an end device identified by its full format (64bits) DevEUI. This interface may be used when a device is associated with a single type of IoT cloud application. Therefore, the payload format may be arbitrary.

This document focuses on the following information about the tunnel mode interface.

- The format of LRC to Application Server messages that encapsulate uplink payload data and associated metadata.

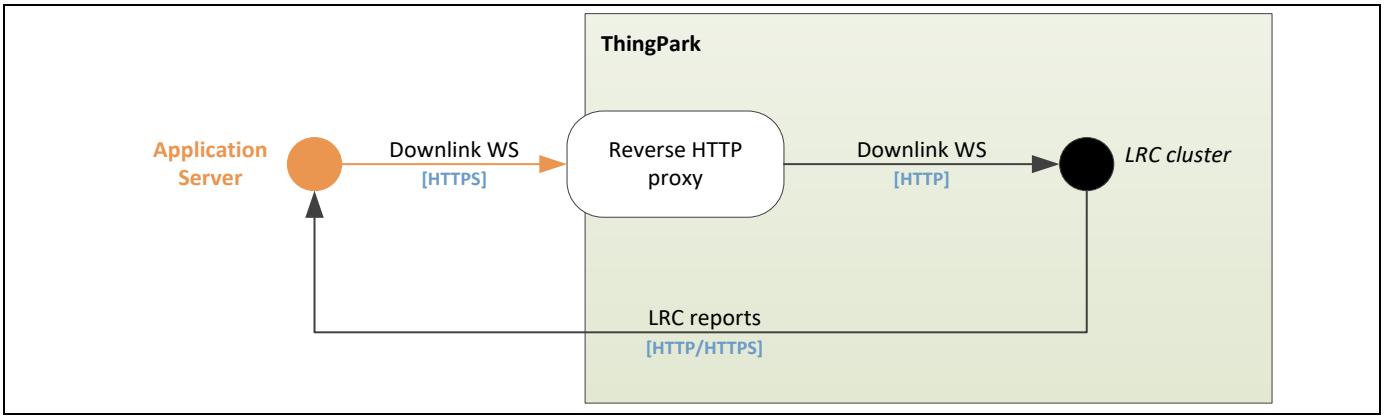
- The format of Application Server to LRC messages that encapsulate downlink payload.

4 CONNECTIVITY

4.1 Uplink HTTP or HTTPS / Downlink HTTPS

The tunneling interface is based on HTTP or HTTPS for reports from the LRC cluster to the Application Server and HTTPS for downlink packets from the Application Server to the LRC cluster.
The destination URL of the LRC reports is defined in the Generic Application configuration.
The LRC reports are transmitted in HTTP/POST or HTTPS/POST requests containing query parameters and an XML or JSON payload.
The confidentiality of the messages is managed by an HTTPS connection.

- The LRC reports HTTPS session is mounted between the LRC cluster and the Application Server.
- The downlink HTTPS session is mounted between the Application Server and the Reverse HTTP proxy in front of the LRC cluster.



The authentication layer relies on a Tunnel Interface Authentication Key shared between the LRC and the Application Server.

- The Application Server uses the Tunnel Interface Authentication Key to generate a signature added to the downlink packets. The LRC uses the signature to verify the identity and the authorization of the Application Server. If the identity or the authorization cannot be verified, the packet is dropped.

- The LRC uses the Tunnel Interface Authentication Key to generate a signature added to the LRC reports. The AS uses the signature to verify the identity of the LRC. If the identity cannot be verified, the Application Server drops the packet.

Note

The Application Server must satisfy the following requirements regarding the LRC reports HTTPS session.

- The Application Server configured in ThingPark application MUST have valid DNS entries, reachable IP addresses and operational HTTP servers.
- HTTP 1.1 is designed to allow pipelining of many requests over a single TCP connection. This feature allows avoiding the latency of TCP SYN/SYN ACK exchange during TCP session setup and therefore lower the perceived request roundtrip. It and also allows to minimize the TLS overhead as the CPU intensive key negotiation happens only during TCP setup. Unfortunately, recent HTTP servers come with default configurations that tend to optimize response to large numbers of clients, and force TCP disconnect after 5 to 15 seconds. This default setup is way too short for an IoT Application Server as it causes TLS full negotiation to happen for each received packet, if the uplink packet interval to Application Server is greater than 5 seconds. **All HTTP servers and proxies used as Application Server for IoT LPWAN networks MUST be configured with a value of at least 30minutes.**
- The Application Server MUST answer to the HTTP request in less than 100 ms. Asynchronous processing of HTTP requests MUST be implemented to guarantee this maximum delay.

5 PARAMETERS FORMAT

5.1 ISO 8601 Timestamps

5.1.1 Timestamps Description

All ISO 8601 timestamps parameters described in this section are mandatory and use the following convention:

`YYYY-MM-DDThh:mm:ss.s+|-hh:mm`

Examples

`2016-08-01T09:06:06.0+02:00`

`2016-11-28T09:06:06.0-04:00`

Where:

- `YYYY`: four-digit year
- `MM`: two-digit month
- `DD`:two-digit day
- `Hh`: two-digit hours (00 through 23)
- `mm`: two-digit minutes (00 through 59)
- `ss`: two-digit seconds (00 through 59)
- `s`: one to three-digit milliseconds (0 through 999)
- `+|-hh:mm` : time zone designator (`+hh:mm` or `-hh:mm`)

5.1.2 Timestamp Encoding When Uplink/Downlink Security Is Activated

When the uplink/downlink security of the Application Server is activated, the timestamp parameter has to be encoded differently in the SHA256 and in the URL of the HTTP request.

Special Characters	In SHA256, use…	In URL, use…
+	+	%2B
-	-	-
:	:	%3A
.	.	.
Examples	2016-08-01T09:06:06.0+02:00 2016-11-28T09:06:06.0-04:00	2016-08-01T09%3A06%3A06.0%2B02%3A00 2016-11-28T09%3A06%3A06.0-04%3A00

6 LRC TO APPLICATION SERVER TUNNEL INTERFACE

This section describes the two reports that are generated from the LRC to an Application Server.

6.1 Uplink Frame Report

6.1.1 Overview

The purpose of this report is to tunnel an uplink packet received from a device.
An uplink packet is reported to the Application Server if it satisfies one of the following conditions:

- The uplink packet contains an applicative payload (FPort > 0)
- If the device implements LoRaWAN® 1.0: the uplink packet does not contain an applicative payload (FPort = 0 or FPort absent) but the ACK bit is set
- If the device implements LoRaWAN® 1.1: the uplink packet does not contain an applicative payload (FPort = 0 or FPort absent) but the ACK bit is set and acknowledges an AFCntDn

In all other cases, the uplink packet is not reported to the Application Server.

The XML or JSON document sent to the Application Server includes the RF metadata corresponding to a maximum of 10 best receiving base stations.

6.1.2 Query Parameters (HTTP only)

The following query parameters generate an uplink frame report.

Parameter	Description
AS_ID	Application Server ID Syntax: STRING
LrnDevEui	Device DevEUI Syntax: STRING (Hexadecimal representation)
LrnFPort	LoRaWAN® port number. Only set if present in the uplink packet. Syntax: NUMBER (Unsigned integer: `0..224`)
LrnInfos	Service profile name used to route the packet Syntax: STRING
Time	ISO 8601 timestamp associated with the generation of the HTTP request by the LRC Syntax: STRING (ISO date/time)

Token	Security token generated by the LRC Syntax: STRING (256 bits hexadecimal)

6.1.3 Document Fields

The following fields (XML elements or JSON keys) describe the uplink frame report document.

Field	Description
ACKbit	`ACKBit` set by the device. `ACKbit` is not filled in the document, if not set in the uplink frame. Syntax: NUMBER (Unsigned integer: `0..1`)
AckRequested	ACK requested indicator (Boolean). Syntax: NUMBER (Unsigned integer: `0..1`)
ADRAckReq	ADR acknowledgment request bit. Syntax: NUMBER (Unsigned integer: `0..1`)
ADRbit	`ADRBit` set by the device. `ADRbit` is not filled in the document, if not set in the uplink frame. Syntax: NUMBER (Unsigned integer: `0..1`)
AFCntDn	The applicative downlink counter to be used for the next downlink frame. See `FCntDn` field for reporting rules. Only applicable to LoRaWAN® 1.1 when the `FPort` > 0. Syntax: NUMBER (32 bits unsigned integer)
BatteryLevel	Battery level reported by the ReportDevStatus. The battery level is reported once a week. Syntax: NUMBER (Unsigned integer: `0..255`)
BatteryTime	Battery level reporting timestamp (ISO 8601). The battery level is reported once a week. Syntax: STRING (ISO date/time)
Channel	LC used by the device. Syntax: STRING (`"LC0".."LC255"`)
ClassBPeriodicity	Class B periodicity in seconds requested by the device. Only applicable to Class B. Syntax: NUMBER (Integer)
ConfAFCntDn	The applicative downlink counter CONFIRMED by the device for this packet. Only applicable to LoRaWAN® 1.1 when the uplink frame acknowledges a previous downlink frame.

Field	Description
	Syntax: NUMBER (32 bits unsigned integer)
CustomerData	JSON customer data set by provisioning: ■ "loc": administrative location (OPTIONAL) • "lat" (latitude) and "lon" (longitude) sub-attributes. ■ "alr": application layer (OPTIONAL) • "pro" (product) and "ver" (version) sub-attributes. XML Syntax: STRING JSON Syntax: OBJECT
CustomerID	Customer ID associated to the ThingPark Enterprise account. Syntax: STRING
DevAddr	Device DevAddr. Syntax: STRING (Hexadecimal representation)
DevEUI	Device DevEUI. Syntax: STRING (Hexadecimal representation)
DevLrrCnt	Number of LRRs which received this packet. Syntax : NUMBER (Unsigned integer)
DevLrrCnt	Number of LRRs which received this packet. Syntax : NUMBER (Unsigned integer)
DevLocTime	Last geolocation timestamp (ISO 8601). Syntax: STRING (ISO date/time)
DevLAT	Last geolocation latitude (GPS coordinate system). Syntax: NUMBER (Float)
DevLON	Last geolocation longitude (GPS coordinate system). Syntax: NUMBER (Float)
DevAlt	Last geolocation altitude (meter). Syntax: NUMBER (Float)
DevLocRadius	Last geolocation horizontal tolerance (meter). Syntax: NUMBER (Float)
DevAltRadius	Last geolocation vertical tolerance (meter). Syntax: NUMBER (Float)
DevNorthVel	Velocity (NORTH axis) expressed in m/s

Field	Description
	Syntax: NUMBER (Float)
DevEastVel	Velocity (EAST axis) expressed in m/s Syntax: NUMBER (Float)
DevLocDilution	Horizontal DOP (dilution of precision) calculated for given mutual position of gateways, used for solution calculation, and device. Syntax: NUMBER (Float)
DevAltDilution	Vertical DOP (dilution of precision) calculated for given mutual position of gateways, used for solution calculation, and device. Syntax: NUMBER (Float)
DriverCfg	Metadata for Driver selection. XML Syntax: STRING JSON Syntax: OBJECT
DynamicClass	LoRaWAN® class currently used by the Device: 'A', 'B' or 'C' Syntax: STRING (Enumerate)
FCntDn	The downlink counter to be used for the next downlink frame. If the downlink queue contains at least one payload encrypted by AS: ■ The reported counter is the highest downlink counter in the downlink queue + 1 Else: ■ The reported counter is the last downlink counter used by the LRC to transmit a downlink + 1 Only applicable to LoRaWAN® 1.0. Syntax: NUMBER (32 bits unsigned integer)
FCntUp	The uplink counter for this packet. Syntax: NUMBER (32 bits unsigned integer)
FPort	LoRaWAN® FPort used by the device for this packet. Only set if present in the uplink packet. Syntax: NUMBER (Unsigned integer: `0..224`)
Frequency	Frequency in MHz of the radio channel used to receive the frame. Syntax: NUMBER (Float)

Field	Description
InstantPER	Instant PER (Packet Error Rate). The instant PER is computed from a sliding window of last packets. Syntax: NUMBER (Float)
Late	Indicates if the packet was queued by the LRR. **Late** is always filled. 0 means that the packet was not queued by the LRR. 1 means that the packet was queued (the LRR queues packets when the connection between the LRR and the LRC is temporarily out of service). Syntax: NUMBER (Unsigned integer: 0..1)
Lrcid	ID of the LRC that processed the packet. Syntax: STRING (Hexadecimal representation)
Lrrid	The ID of the LRR that received the packet with the best SNR. This LRR is flagged as "best LRR". Syntax: STRING (Hexadecimal representation)
LrrLAT **LrrLON**	LAT and LON of the best LRR. Syntax: NUMBER (Float)
LrrRSSI	RSSI measured by the best LRR. Syntax: NUMBER (Float)
LrrSNR	SNR measured by the best LRR. Syntax: NUMBER (Float)
Lrrs/Lrr/Chain	The LRR antenna chain used to receive the packet. Syntax: NUMBER (Unsigned integer)
Lrrs/Lrr/Lrrid	LRR ID associated to this <Lrr> field. See Notes. Syntax: STRING (Hexadecimal representation)
Lrrs/Lrr/LrrESP	ESP measured by the LRR associated to this <Lrr> field element. See Notes. Syntax: NUMBER (Float)
Lrrs/Lrr/LrrRSSI	RSSI measured by the LRR associated to this <Lrr> field. See Notes. Syntax: NUMBER (Float)
Lrrs/Lrr/LrrSNR	SNR measured by the LRR associated to this <Lrr> field. See Notes. Syntax: NUMBER (Float)
Margin	Margin reported by the ReportDevStatus. The margin is reported once a week.

Field	Description
	Margin is a signed integer of 6 bits with a minimum value of -32 and a maximum value of 31. It is reported as a 6 bits unsigned integer in `Margin` field. The following formula must be applied to get the actual Margin value: ■ If `Margin < 32`: the value is kept as is. ■ Else (`Margin >= 32`): the actual Margin value is `Margin - 64`. Syntax: NUMBER (Unsigned integer: `0..63`)
MeanPER	Mean PER (Packet Error Rate). Syntax: NUMBER (Float)
mic_hex	MIC in hexadecimal ASCII format. Syntax: STRING (Hexadecimal representation)
MType	LoRaWAN® MType of the packet. Syntax: NUMBER (Unsigned integer: `0..7`)
NbTrans	The number of transmissions for each uplink message requested by the LRC, according to ADR algorithm and validated by the device through a **LinkADRAns** MAC command. If no **LinkADRAns** have been yet validated by the device, NbTrans = 1. Syntax: NUMBER (Unsigned Integer)
NwGeolocAlgo	Network geolocation algorithm configured for the device: ■ 0 = Time Difference Of Arrival (TDOA): The algorithm is based on an uplink frame time of arrival and received signal characteristics at the receiving gateways. ■ 1 = Received Signal Strength Indicator (RSSI): The algorithm is based on an uplink frame received signal strength of at the receiving gateways. ■ 2 = Both: The calculation combines both TDOA and RSSI algorithms: RSSI algorithm is used if TDOA algorithm does not provide a valid geolocation. `NwGeolocAlgo` is only filled if a geolocation is reported. Syntax: NUMBER (Unsigned integer: `0..2`)
NwGeolocAlgoUsed	Network geolocation algorithm used by the geolocation solver: ■ 0 = Time Difference Of Arrival (TDOA): The algorithm is based on an uplink frame time of arrival and received signal characteristics at the receiving gateways.

Field	Description
	■ 1 = Received Signal Strength Indicator (RSSI): The algorithm is based on an uplink frame received signal strength of at the receiving gateways. `NwGeolocAlgoUsed` is only filled if a geolocation is reported. Syntax: NUMBER (Unsigned integer: `0..1`)
payload_hex	LoRaWAN® payload in hexadecimal ASCII format. Syntax: STRING (Hexadecimal representation)
rawMacCommands	Hexadecimal ASCII dump of the LoRaWAN® MAC option or LoRaWAN® MAC payload (port=0). Syntax: STRING (Hexadecimal representation)
SpFact	SF used by the device. Syntax: NUMBER (Unsigned integer: `7..12`)
SubBand	Sub-band used by the device. Syntax: STRING
Time	LRR Timestamp for the packet. Syntax: STRING (ISO date/time)
TxPower	The transmission power of the device (in dBm), computed by the LRC based on ADR algorithm and validated by the device through a **LinkADRAns**. If no **LinkADRAns** have been yet validated by the device, the device boot parameter is used instead. Syntax: NUMBER (Float)
UplinkDC	The aggregate occupancy rate of the RF logical channel associated with this uplink frame, as estimated at the time of reception. Syntax: NUMBER (Float)
UplinkDCSubBand	The aggregate occupancy rate of the RF sub-band associated with this uplink frame, as estimated at the time of reception. Syntax: NUMBER (Float)

Note

- The <`Lrrs`> fields contain up to 10 LRRs which reported the packet before the expiration of the 250 ms deduplication window. If the packet was received by more than 10 LRRs before the expiration of the 250 ms deduplication window, only the 10 LRRs with the best SNR are reported.

- The <DevLocTime>, <DevLAT>, <DevLON>, <DevAlt>, <DevLocRadius>, <DevAltRadius>, <DevLocDilution> and <DevAltDilution> fields are only filled if the network geolocation mode is activated for the device, and if a geolocation has been resolved based on previous uplink frames. When a geolocation is provided, the <DevLocTime>, <DevLAT>, <DevLON> fields are always filled. Depending on the geolocation resolution output, the other fields may not be available.
- The geolocation data reported for FCntUp=N is that which was resolved from previous uplink frames (At the time of transmitting the current report for FCntUp=N, the solver is still processing FCntUp=N information). Refer to *6.3 Location Report* for asynchronous reporting of the geolocation associated to the current uplink frame (current FCntUp) as soon as available from the location server.

6.1.4 Sample of HTTP Request

Here is a sample of an HTTP request for an uplink frame report in XML format, where <as-url> is the destination URL configured in the Generic Application configuration.

Note: In an URL, the + and : characters must be escaped.

```
>> POST <as-url>?LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.s
ample.com&Time=2016-01-11T14%3A11%3A11.333%2B02%3A00&Token=fd0b0b00464aa798a59282d64eaa70813
e33bff87682880db49638569d096aad

<?xml version="1.0" encoding="UTF-8"?>
<DevEUI_uplink xmlns="http://uri.actility.com/lora">
   <Time>2015-07-09T16:06:38.49+02:00</Time>
   <DevEUI>000000000F1D8693</DevEUI>
   <FPort>2</FPort>
   <FCntUp>7011</FCntUp>
   <ADRbit>1</ADRbit>
   <ACKbit>1</ACKbit>
   <MType>4</MType>
   <FCntDn>11</FCntDn>
   <payload_hex>0027bd00</payload_hex>
   <mic_hex>38e7a3b9</mic_hex>
   <Lrcid>00000065</Lrcid>
   <LrrRSSI>-60.000000</LrrRSSI>
   <LrrSNR>9.750000</LrrSNR>
   <SpFact>7</SpFact>
   <SubBand>G1</SubBand>
   <Channel>LC2</Channel>
   <DevLrrCnt>2</DevLrrCnt>
   <Lrrid>08040059</Lrrid>
   <Late>0</Late>
   <LrrLAT>48.874931</LrrLAT>
   <LrrLON>2.333673</LrrLON>
   <Lrrs>
       <Lrr>
           <Lrrid>08040059</Lrrid>
           <LrrRSSI>-60.000000</LrrRSSI>
           <LrrSNR>9.750000</LrrSNR>
           <LrrESP>-59.000000</LrrESP>
       </Lrr>
       <Lrr>
           <Lrrid>33d13a41</Lrrid>
           <LrrRSSI>-73.000000</LrrRSSI>
           <LrrSNR>9.750000</LrrSNR>
           <LrrESP>-72.000000</LrrESP>
       </Lrr>
   </Lrrs>
   <CustomerID>100000507</CustomerID>
   <CustomerData>
      {"loc":{"lat":"43.58081","lon":"1.4421667"},"alr":{"pro":"STL","ver":"1"}}
   </CustomerData>
   <InstantPER>0.02</InstantPER>
   <MeanPER>0.02</MeanPER>
   <DevAddr>0405F519</DevAddr>
   <UplinkDC>0.001</UplinkDC>
   <UplinkDCSubBand>0.009</UplinkDCSubBand>
   <DevLocTime>2015-01-27T10:00:43.336+01:00</DevLocTime>
   <DevLAT>10.11212</DevLAT>
   <DevLON>7.44464</DevLON>
   <DevAlt>50</DevAlt>
   <DevLocRadius>100</DevLocRadius>
   <DevAltRadius>50</DevAltRadius>
   <DevNorthVel>1.0</DevNorthVel>
   <DevEastVel>1.0</DevEastVel>
   <NwGeolocAlgo>0</NwGeolocAlgo>
   <NwGeolocAlgoUsed>0</NwGeolocAlgoUsed>
   <Frequency>868.3</Frequency>
   <DynamicClass>B</DynamicClass>
   <ClassBPeriodicity>128</ClassBPeriodicity>
   <DriverCfg>
      {"mod":{"pId":"abeeway","mId":"micro-tracker","ver":"3"},"app":{"pId":"abeeway","mId":
"asset-tracker","ver":"1"}}
   </DriverCfg>
</DevEUI_uplink>
```

6.1.5 Payload Encryption

The payload is always provided decrypted to the application.

6.2 Downlink Frame Sent Report

6.2.1 Overview

The main purpose of this report is to give the effective RF transmission status of a downlink frame transmission initiated by an Application Server. The RF transmission status of downlink frames initiated by the LRC itself are not reported to the Application Server.
Effective RF transmission only ensures that the downlink frame has effectively been transmitted by an LRR base station. It does not guarantee that the device has received it. Reports that need end-to-end delivery reports should be sent in Confirmed mode or use an application layer mechanism.

6.2.2 Query Parameters (HTTP only)

The following query parameters generate a report of downlink frame sent.

Parameter	Description
AS_ID	Application Server ID Syntax: STRING
LrnDevEui	Device DevEUI Syntax: STRING (Hexadecimal representation)
LrnFPort	LoRaWAN® port number Syntax: NUMBER (Unsigned integer: `1..224`)
LrnInfos	Service profile name used to route the packet Syntax: STRING
Time	ISO 8601 timestamp associated to the generation of the HTTP request by the LRC Syntax: STRING (ISO date/time)
Token	Security token generated by the LRC Syntax: STRING (256 bits hexadecimal)

6.2.3 Document Fields

The following fields (XML elements or JSON keys) describe the downlink frame sent report document.

Field	Description
AckRequested	ACK requested indicator (Boolean). Syntax: NUMBER (Unsigned integer: `0..1`)
ADRAckReq	ADR acknowledgment request bit. Syntax: NUMBER (Unsigned integer: `0..1`)
AFCntDn	The applicative downlink counter for this packet. Only applicable to LoRaWAN® 1.1 when the `FPort` > 0. Syntax: NUMBER (32 bits unsigned integer)
Channel	LC used by the device. Syntax: STRING (`"LC0"..."LC255"`)
CorrelationID	ID used to correlate the downlink frame sent report with the unicast downlink frame previously submitted. Only reported when `CorrelationID` has been provided in downlink frame. Syntax: STRING (64 bits hexadecimal)
CustomerData	JSON customer data set by provisioning: ■ `"loc"`: administrative location (OPTIONAL) • `"lat"` (latitude) and `"lon"` (longitude) sub-attributes. ■ `"alr"`: application layer (OPTIONAL) • `"pro"` (product) and `"ver"` (version) sub-attributes. XML Syntax: STRING JSON Syntax: OBJECT
CustomerID	Customer ID associated to the ThingPark Enterprise account. Syntax: STRING
DeliveryStatus	RF transmission status: ■ `1`: Downlink frame was sent over the air (either on RX1 or RX2). This means that the downlink frame was transmitted over the air by the LRR. But the downlink frame may not have been received by the device. ■ `0`: Downlink frame was not sent over the air (neither on RX1 nor RX2). The downlink frame is not retransmitted by the network server. Accordingly, the

Field	Description
	downlink frame will have to be reinitiated by the Application Server. Syntax: NUMBER (Unsigned integer: `0..1`)
`DeliveryFailedCause1`	The over the air delivery error cause for RX1 downlink slot. ■ **Class A device**: Transmission slot busy on RX1: • `A0`: "Radio stopped" • `A1`: "Downlink radio stopped" • `A3`: "Radio busy" • `A4`: "Listen before talk" • `A5`: "Radio board error" ■ **Class A device**: Received too late for RX1: • `B0`: "Too late for RX1" ■ **Class A device**: LRC selects RX2: • `C0`: "LRC selected RX2" ■ **Class A device**: DC or Gateway constraint on RX1: • `D0`: "Duty cycle constraint detected by LRR" • `DA`: "Duty cycle constraint detected by LRC" • `DB`: "Max dwell time constraint detected by the LRC" • `DE`: "DC not allowed by the peering operator" `DeliveryFailedCause1` is set to `00` when no error occurs on RX1. Syntax: STRING (Hexadecimal representation)
`DeliveryFailedCause2`	The over the air delivery error cause for RX2 downlink slot. ■ **Class A device**: Transmission slot busy on RX2: • `A0`: "Radio stopped" • `A1`: "Downlink radio stopped" • `A3`: "Radio busy" • `A4`: "Listen before talk" • `A5`: "Radio board error" ■ **Class A device**: Received too late for RX2: • `B0`: "Too late for RX2" ■ **Class A device**: DC or Gateway constraint on RX2: • `D0`: "Duty cycle constraint detected by LRR" • `DA`: "Duty cycle constraint detected by LRC" • `DB`: "Max dwell time constraint detected by the LRC" • `DE`: "DC not allowed by the peering operator" ■ **Class C device**: Frame expired before transmission:

Field	Description
	• `E0`: "Max delay for Class C" (60 seconds) `DeliveryFailedCause2` is set to `00` when no error occurs on RX2. Syntax: STRING (Hexadecimal representation)
`DeliveryFailedCause3`	The over the air delivery error cause for downlink ping slot. ■ **Class B device**: Transmission slot busy on ping slot: • `A0`: "Radio stopped" • `A1`: "Downlink radio stopped" • `A2`: "Ping slot not available" • `A3`: "Radio busy" • `A4`: "Listen before talk" • `A5`: "Radio board error" ■ **Class B device**: Received too late for ping slot: • `B0`: "Too late for ping slot" ■ **Class B device**: DC or Gateway constraint on ping slot: • `D0`: "Duty cycle constraint detected by LRR" • `DA`: "Duty cycle constraint detected by LRC" • `DB`: "Max dwell time constraint detected by the LRC" • `DC`: "No GPS-synchronized LRR detected by the LRC" • `DD`: "No LRR connected detected by the LRC" • `DF`: "Wrong NetID" `DeliveryFailedCause3` is set to `00` when no error occurs on ping slot. Syntax: STRING (Hexadecimal representation)
`DevAddr`	Device Addr. Syntax: STRING (Hexadecimal representation)
`DevEUI`	Device DevEUI. Syntax: STRING (Hexadecimal representation)
`FCntDn`	The downlink counter for this packet. Only applicable to LoRaWAN® 1.0. Syntax: NUMBER (32 bits unsigned integer)
`FCntUp`	The last uplink counter from the device. Syntax: NUMBER (32 bits unsigned integer)

Field	Description
FPending	Pending downlink frame indicator. Syntax: NUMBER (Unsigned integer: `0..1`)
FPort	LoRaWAN® `FPort` used by the device for this packet. Syntax: NUMBER (Unsigned integer: `1..224`)
Frequency	Frequency in MHz of the radio channel used to send the frame. Syntax: NUMBER (Float)
Lrcid	ID of the LRC that processed the packet. Syntax: STRING (Hexadecimal representation)
Lrrid	ID of the LRR used to send the packet. Syntax: STRING (Hexadecimal representation)
rawMacCommands	Hexadecimal ASCII dump of the LoRaWAN® MAC option or LoRaWAN® MAC payload (port=0). Syntax: STRING (Hexadecimal representation)
SpFact	RX1 SF selected by the LRC. Syntax: NUMBER (Unsigned integer: `7..12`)
SubBand	Sub-band used by the device. Syntax: STRING
Time	LRR Timestamp for the packet. Syntax: STRING (ISO date/time)
TransmissionSlot	Slot used for downlink frame transmission: 0 (Unknown), 1 (RX1), 2 (RX2) or 3 (Ping Slot) Syntax: NUMBER (Unsigned integer: `0..3`)

6.2.4 Sample of HTTP Request

Here is a sample of an HTTP request for a downlink frame sent report in XML format, where <as-url> is the destination URL configured in the Generic Application configuration.

```
>> POST <as-url>?LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.s
ample.com&Time=2016-01-11T14%3A22%3A22.333%2B02%3A00&Token=c85e44b8c386053962fd22be4b9728d77
0b4c767952f1a6a741120be300776e7

<?xml version="1.0" encoding="UTF-8"?>
<DevEUI_downlink_Sent xmlns="http://uri.actility.com/lora">
   <Time>2015-01-27T10:00:46.336+01:00</Time>
   <DevEUI>000000000F1D8693</DevEUI>
   <FPort>2</FPort>
   <FCntUp>7022</FCntUp>
```

```
    <FCntDn>22</FCntDn>
    <Lrcid>00000000</Lrcid>
    <SpFact>7</SpFact>
    <SubBand>G1</SubBand>
    <Channel>LC3</Channel>
    <Lrrid>0000000a</Lrrid>
    <DeliveryStatus>1</DeliveryStatus>
    <DeliveryFailedCause1>A3</DeliveryFailedCause1>
    <DeliveryFailedCause2>00</DeliveryFailedCause2>
    <CustomerID>100000507</CustomerID>
    <CustomerData>
       {"loc":{"lat":"43.58081","lon":"1.4421667"},"alr":{"pro":"STL","ver":"1"}}
    </CustomerData>
    <TransmissionSlot>1</TransmissionSlot>
    <Frequency>868.5</Frequency>
    <CorrelationID>4434704901C7450B</CorrelationID>
</DevEUI_downlink_Sent>
```

6.3 Location Report

6.3.1 Overview

The purpose of the location report is to inform, asynchronously, about the geolocation data associated to the current uplink frame (current `FCntUp`), as soon as available from the location server.

The location message is only reported if the network geolocation mode is activated for the device, and if a new geolocation has been resolved.

6.3.2 Query Parameters (HTTP only)

The following query parameters are defined to generate a location report.

Parameter	Description
AS_ID	Application Server ID Syntax: STRING
LrnDevEui	Device DevEUI. Syntax: STRING (Hexadecimal representation)
LrnInfos	Service profile name used to route the packet. Syntax: STRING
Time	ISO 8601 timestamp associated to the generation of the HTTP request by the LRC Syntax: STRING (ISO date/time)
Token	Security token generated by the LRC Syntax: STRING (256 bits hexadecimal)

6.3.3 Document Fields

The following fields (XML elements or JSON keys) describe the location report document.

Field	Description
`CustomerID`	Customer ID associated to the ThingPark Enterprise account. Syntax: STRING
`DevAddr`	Device DevAddr. Syntax: STRING (Hexadecimal representation)
`DevEUI`	Device DevEUI. Syntax: STRING (Hexadecimal representation)
`DevLocTime`	Last geolocation timestamp (ISO 8601). Syntax: STRING (ISO date/time)
`DevLAT`	Last geolocation latitude (GPS coordinate system). Syntax: NUMBER (Float)
`DevLON`	Last geolocation longitude (GPS coordinate system). Syntax: NUMBER (Float)
`DevAlt`	Last geolocation altitude (meter). Syntax: NUMBER (Float)
`DevLocRadius`	Last geolocation horizontal tolerance (meter). Syntax: NUMBER (Float)
`DevAltRadius`	Last geolocation vertical tolerance (meter). Syntax: NUMBER (Float)
`DevNorthVel`	Velocity (NORTH axis) expressed in m/s Syntax: NUMBER (Float)
`DevEastVel`	Velocity (EAST axis) expressed in m/s Syntax: NUMBER (Float)
`DevLocDilution`	Horizontal DOP (dilution of precision) calculated for given mutual position of gateways, used for solution calculation, and device. Syntax: NUMBER (Float)
`DevAltDilution`	Vertical DOP (dilution of precision) calculated for given mutual position of gateways, used for solution calculation, and device. Syntax: NUMBER (Float)
`DevUlFCntUpUsed`	The uplink counter used for this location resolution.

Field	Description
	Syntax: NUMBER (32 bits unsigned integer)
Lrcid	ID of the LRC that processed the geolocation. Syntax: STRING (Hexadecimal representation)
NwGeolocAlgo	Network geolocation algorithm configured for the device: ■ 0 = Time Difference Of Arrival (TDOA): The algorithm is based on an uplink frame time of arrival and received signal characteristics at the receiving gateways. ■ 1 = Received Signal Strength Indicator (RSSI): The algorithm is based on an uplink frame received signal strength of at the receiving gateways. ■ 2 = Both: The calculation combines both TDOA and RSSI algorithms: RSSI algorithm is used if TDOA algorithm does not provide a valid geolocation. Syntax: NUMBER (Unsigned integer: `0..2`)
NwGeolocAlgoUsed	Network geolocation algorithm used by the geolocation solver: ■ 0 = Time Difference Of Arrival (TDOA): The algorithm is based on an uplink frame time of arrival and received signal characteristics at the receiving gateways. ■ 1 = Received Signal Strength Indicator (RSSI): The algorithm is based on an uplink frame received signal strength of at the receiving gateways. Syntax: NUMBER (Unsigned integer: `0..1`)
Time	LRC Timestamp for the geolocation. Syntax: STRING (ISO date/time)

6.3.4 Sample of HTTP Request

Here is a sample of an HTTP request for a location report in XML format, where `<as-url>` is the destination URL configured in the Generic Application configuration.

Note: In an URL, the `+` and `:` characters must be escaped.

```
>> POST <as-url>?LrnDevEui=000000000f1d8693&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&T
ime=2016-01-11T14%3A22%3A22.333%2B02%3A00&Token=2cb9c53e6bd861fcf01cf47d14a93cb64a264da5756c
ced63148da729207e6b7

<?xml version="1.0" encoding="UTF-8"?>
<DevEUI_location xmlns="http://uri.actility.com/lora">
   <Time>2015-01-27T10:00.123:46+01:00</Time>
   <DevEUI>000000000F1D8693</DevEUI>
   <DevAddr>0F1D8693</DevAddr>
   <Lrcid>00000000</Lrcid>
   <DevLocTime>2015-01-27T10:00:43.336+01:00</DevLocTime>
   <DevLAT>10.11212</DevLAT>
   <DevLON>7.44464</DevLON>
   <DevAlt>50</DevAlt>
```

```
    <DevLocRadius>100</DevLocRadius>
    <DevAltRadius>50</DevAltRadius>
    <DevNorthVel>1.0</DevNorthVel>
    <DevEastVel>1.0</DevEastVel>
    <DevUlFCntUpUsed>7011</DevUlFCntUpUsed>
    <CustomerID>100000507</CustomerID>
    <NwGeolocAlgo>0</NwGeolocAlgo>
    <NwGeolocAlgoUsed>0</NwGeolocAlgoUsed>
</DevEUI_location>
```

6.4 Notification Report

6.4.1 Overview

The purpose of the notification report is to notify the following events:

- **Device reset**: A device reset has been detected by the LRC. A device reset is detected in the following cases:
  - **OTAA device administrative reset via Device Manager API**
    - *LoRaWAN® 1.0*: The notification report is triggered by the administrative reset request and contains an LRC timestamp. The FCntDn is reset to 0 in this report.
    - *LoRaWAN® 1.1*: The notification report is triggered by the administrative reset request and contains an LRC timestamp. The AFCntDn is reset to 0 in this report.
  - **ABP device reset detection on an uplink frame**
    - *LoRaWAN® 1.0 when ABP automatic reset is allowed*: The notification report is triggered by the uplink frame for which the FCntUp reset has been detected and contains the LRR timestamp of this join-request. The FCntDn is reset to 0 in this report.
    - *LoRaWAN® 1.1*: The notification report is triggered by the uplink frame containing the ResetInd MAC command and contains the LRR timestamp of this uplink frame. The AFCntDn is NOT reset to 0 in this report.
  - **ABP device administrative reset via Device Manager API**
    - *LoRaWAN® 1.0*: The notification report is triggered by the administrative reset request and contains an LRC timestamp. The FCntDn is reset to 0 in this report.
    - *LoRaWAN® 1.1*: The notification report is triggered by the administrative reset request and contains an LRC timestamp. The AFCntDn is reset to 0 in this report.
- **Successful Join procedure**: After sending the Join-accept message, the new AppSKey is reported to the Application Server. A successful join procedure is detected in the following cases:
  - **Successful Join procedure on first join-request**
    - *LoRaWAN® 1.0*: The notification report is triggered by the join-request and contains the LRR timestamp of this join-request. The FCntDn is reset to 0 in this report.

- *LoRaWAN® 1.1*: The notification report is triggered by the uplink frame containing the ReKeyInd MAC command and contains the LRR timestamp of this uplink frame. The AFCntDn is reset to 0 in this report.

- **Successful Join procedure on subsequent join-request**
  - *LoRaWAN® 1.0 with DevNonce counter-based*: The notification report is triggered by the join-request and contains the LRR timestamp of this join-request. The FCntDn is reset to 0 in this report.
  - *LoRaWAN® 1.0 without DevNonce counter-based*: The notification report is triggered by the next uplink frame validating the join procedure and contains the LRR timestamp of this uplink frame. The FCntDn is reset to 0 in this report.
  - *LoRaWAN® 1.1*: The notification report is triggered by the uplink frame containing the ReKeyInd MAC command and contains the LRR timestamp of this uplink frame. The AFCntDn is reset to 0 in this report.

- **Battery and Margin**: The LRC received a DevStatusAns MAC command including Battery and Margin information. The notification report is only triggered when the uplink frame containing the DevStatusAns MAC command does not contain an applicative payload. When the uplink frame contains an applicative payload, the Battery and Margin information is included in the uplink frame report.

6.4.2 Query Parameters (HTTP only)

The following query parameters are defined to generate a notification report.

Parameter	Description
`AS_ID`	Application Server ID Syntax: STRING
`LrnDevEui`	Device DevEUI Syntax: STRING (Hexadecimal representation)
`LrnInfos`	Service profile name used to route the packet Syntax: STRING
`Time`	ISO 8601 timestamp associated to the generation of the HTTP request by the LRC Syntax: STRING (ISO date/time)
`Token`	Security token generated by the LRC Syntax: STRING (256 bits hexadecimal)

6.4.3 Document Fields

The following fields (XML elements or JSON keys) describe the notification report document.

Field	Description
AFCntDn	The applicative downlink counter to be used for the next downlink frame. Only applicable to LoRaWAN® 1.1. Syntax: NUMBER (32 bits unsigned integer)
CustomerID	Customer ID associated to the ThingPark Enterprise account. Syntax: STRING
DevAddr	Device DevAddr. Syntax: STRING (Hexadecimal representation)
DevEUI	Device DevEUI. Syntax: STRING (Hexadecimal representation)
FCntDn	The downlink counter to be used for the next downlink frame. Only applicable to LoRaWAN® 1.0. Syntax: NUMBER (32 bits unsigned integer)
Lrcid	ID of the LRC that reported the notification. Syntax: STRING (Hexadecimal representation)
Time	If the notification is triggered by an uplink frame: - Uplink frame timestamp as reported by the LRR. Else: - LRC Timestamp for the notification. Syntax: STRING (ISO date/time)
Type	Type of notification: ■ reset: Device reset ■ join: Successful Join procedure ■ devstatusans: Battery and Margin Syntax: STRING (enum)
Var1	Notification variable 1 The content depends on notification type: ■ reset: type of reset ○ automatic_reset: ABP automatic reset ○ admin_reset: OTAA/ABP administrative reset ■ join: AppSKey encrypted with AS transport key ■ devstatusans: Battery (0..255)

Field	Description
	Syntax: STRING
Var2	Notification variable 2 The content depends on notification type: ■ reset: n/a ■ join: n/a ■ devstatusans: Margin (-32..31) Syntax: STRING

6.4.4 Sample of HTTP Request

Here is a sample of an HTTP request for a notification report in XML format, where `<as-url>` is the destination URL configured in the Generic Application configuration.

Note: In an URL, the + and : characters must be escaped.

```
>> POST <as-url>?LrnDevEui=000000000F1D8693&&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&
Time=2019-01-15T11%3A31%3A19.588%2B02%3A00&Token=2cb9c53e6bd861fcf01cf47d14a93cb64a264da5756
cced63148da729207e6b7

<?xml version="1.0" encoding="UTF-8"?>
<DevEUI_notification xmlns="http://uri.actility.com/lora">
   <Time>2019-01-15T11:31:19.588+02:00</Time>
   <DevEUI>F03D291E001001C9</DevEUI>
   <Lrcid>00000602</Lrcid>
   <DevAddr>007E074F</DevAddr>
   <Type>reset</Type>
   <Var1>admin_reset</Var1>
   <CustomerID>100000507</CustomerID>
   <FCntDn>0</FCntDn>
</DevEUI_notification>
```

6.5 LRC Authentication of Reports

This section describes the LRC authentication of the LRC reports.

6.5.1 Securing LRC to AS Frame Principles

Securing LRC to AS frame is implemented in four steps.

- The LRC adds the AS ID and the generation timestamp in the message.
- Then, the LRC adds a security token to sign the message based on a pre-shared Tunnel Interface Authentication Key.
- When the AS receives a message, the AS re-computes the security token.
- If the re-computed security token matches the security token provided by the LRC, then the AS can trust the message and process it accordingly.

The AS ID and the Tunnel Interface Authentication Key are part of the Generic Application configuration associated with the device.
For more information about the Tunnel Interface Authentication Key, see *4 Connectivity*.

6.5.2 Token Verification Details

The token must be verified by the Application Server as follows.

1. The Application Server retrieves the *<query-parameters>* WITHOUT the Token QP (Query parameters including the `AS_ID` and the `Time`):

   - For an uplink frame (based on the example given in *6.1 Uplink Frame*)
     An example of *<query parameters>* is:
     `LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:11:11.333+02:00`

   - For a downlink frame sent report (based on the example given in *6.2 Downlink Frame Sent Report*)
     An example of *<query parameters>* is:
     `LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:22:22.333+02:00`

   - For a location report (based on the example given in *6.3 Location Report*)
     An example of *<query parameters>* is:
     `LrnDevEui=000000000f1d8693&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:22:22.333+02:00`

   - For a notification report (based on the example given in *6.4 Notification Report*)
     An example of *<query parameters>* is:
     `LrnDevEui=000000000F1D8693&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2019-01-15T11:31:19.588+02:00`

2. The Application Server builds the *<body-elements>* as the concatenation, without separator, of the following values:

   - For an uplink frame (extract from the `<DevEUI_uplink>` body): `CustomerID`, `DevEUI`, `FPort`, `FCntUp`, `payload_hex`
     An example of *<body-elements>* is: `1000005070000000000F1D8693270110027bd00`

   - For a downlink frame sent report (extracted from the `<DevEUI_downlink_sent>` body): `CustomerID`, `DevEUI`, `FPort`, `FCntDn`
     An example of *<body-elements>* is: `1000005070000000000F1D8693222`

   - For a location report (extracted from the `<DevEUI_location>` body): `CustomerID`, `DevEUI`
     An example of *<body-elements>* is: `1000005070000000000f1d8693`

   - For a notification report (extracted from the `<DevEUI_notification>` body): `CustomerID`, `DevEUI`
     An example of *<body-elements>* is: `1000005070000000000F1D8693`

3. The Application Server re-computes the *<token>* as:
   SHA-256 (<body-elements><query-parameters><LrcAsKey>)

   - For an uplink frame
     An example of *<token>* is:
     `SHA-256(1000005070000000000F1D8693270110027bd00LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:11:11.333+02:0046ab678cd45df4a4e4b375eacd096acc)`

- For a downlink frame sent report
  An example of *<token>* is:
  `SHA-256(100000507000000000F1D8693222LrnDevEui=000000000F1D8693&LrnFPort=2&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:22:22.333+02:0046ab678cd45df4a4e4b375eacd096acc)`

- For a location report
  An example of *<token>* is:
  `SHA-256(100000507000000000f1d8693LrnDevEui=000000000f1d8693&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2016-01-11T14:22:22.333+02:0046ab678cd45df4a4e4b375eacd096acc)`

- For a notification report
  An example of *<token>* is:
  `SHA-256(100000507000000000F1D8693LrnDevEui=000000000F1D8693&LrnInfos=UPHTTP_LAB_LORA&AS_ID=app1.sample.com&Time=2019-01-15T11:31:19.588+02:0046ab678cd45df4a4e4b375eacd096acc)`

  Where:
  `46ab678cd45df4a4e4b375eacd096acc` is the 128 bits pre-shared Tunnel Interface Authentication Key (lower case hexadecimal string representation) between the Application Server and the LRC as defined in the Generic Application configuration.

4. The *<token>* is encoded as a hexadecimal string AND can be compared to the *<token>* provided by the LRC in the <*query parameters*> line of the HTTP request.

   - For an uplink frame
     An example of *<encrypted-token>* is:
     `fd0b0b00464aa798a59282d64eaa70813e33bff87682880db49638569d096aad`

   - For a downlink frame sent report
     An example of *<encrypted-token>* is:
     `c85e44b8c386053962fd22be4b9728d770b4c767952f1a6a741120be300776e7`

   - For a location report
     An example of *<encrypted-token>* is:
     `76905fbe56c6221361a65ffd979f615309c9272679d3150c07c0739969b6395e`

   - For a notification report
     An example of *<encrypted-token>* is:
     `1f3b419c491e4f302814b98d007b1b97873aec1c42ff03a32a02a1f1a9b7e792`

6.6 XML or JSON Encoding

6.6.1 Setting JSON Format

By default, the message of LRC reports is encoded in JSON format by the LRC. It can be set to JSON untyped or XML in the Generic Application configuration.

Information elements in the XML document (as defined in *6.1 Uplink Frame Report*) can be mapped one-to-one with information elements in the JSON document.

6.6.2 Sample of a JSON Document

Here is a sample of an uplink JSON document.


```
{
  "DevEUI_uplink": {
    "-xmlns": "http://uri.actility.com/lora",
    "Time": "2015-07-09T16:06:38.49+02:00",
    "DevEUI": "000000000F1D8693",
    "FPort": "2",
    "FCntUp": "7011",
    "ADRbit": "1",
    "ACKbit": "1",
    "MType": "4",
    "FCntDn": "11",
    "payload_hex": "0027bd00",
    "mic_hex": "38e7a3b9",
    "Lrcid": "00000065",
    "LrrRSSI": "-60.000000",
    "LrrSNR": "9.750000",
    "SpFact": "7",
    "SubBand": "G1",
    "Channel": "LC2",
    "DevLrrCnt": "2",
    "Lrrid": "08040059",
    "Late": "0",
    "LrrLAT": "48.874931",
    "LrrLON": "2.333673",
    "Lrrs": {
      "Lrr": [
        {
          "Lrrid": "08040059",
          "LrrRSSI": "-60.000000",
          "LrrSNR": "9.750000",
          "LrrESP": "-59.000000"
        },
        {
          "Lrrid": "33d13a41",
          "LrrRSSI": "-73.000000",
          "LrrSNR": "9.750000",
          "LrrESP": "-72.000000"
        }
      ]
    },
    "CustomerID": "100000507",
    "CustomerData": {
      "loc": {
        "lat": "43.58081",
        "lon": "1.4421667"
      },
      "alr": {
        "pro": "STL",
        "ver": "1"
      }
    },
    "InstantPER": "0.02",
    "MeanPER": "0.02",
    "DevAddr": "0405F519",
    "UplinkDC": "0.001",
    "UplinkDCSubBand": "0.009"
  }
}
```

7 APPLICATION SERVER TO LRC TUNNEL INTERFACE

This section describes how the downlinks are sent from the AS to a device.

Notes

- **For Class A devices**, the LRC queues up to five downlink frames for each device. The AS may decide to flush the downlink queue, using the field "`FlushDownlinkQueue`" in the downlink frame query parameters. Additionally, when the downlink queue contains payloads encrypted by AS, it is automatically flushed when an OTAA device rejoins or an ABP device reset (automatic or administrative) is detected. This is required as encrypted payloads are no more valid at this stage: AppSKey is renewed for OTAA devices and FCntDn sequence is reset.
- **For Class B devices**, the LRC forwards the downlink frames to the downlink best-LRR together with the available pingslots for that device in the current and the next beacon period, so that the best-LRR can have sufficient scheduling occasions to re-attempt retransmission of the downlink frame based on its local context (LBT, busy modem …etc). To ensure in-sequence delivery(1) of the downlink frames, the AS should implement a flow control mechanism to send downlink frames one-by-one as follows:
  - Send a downlink frame (FCntDn X)
  - Wait for Downlink Frame Sent Report
    - If SUCCESS: send the next downlink frame (FCntDn X+1)
    - If failed: it is up to AS to decide if a retransmission is required.
- **For Class C devices**, the LRC immediately forwards the downlink frames to the downlink best-LRR without any queuing. In case of transmission issues (resource congestion due to modem business), the best-LRR can autonomously re-attempt packet transmission up to 60s. To ensure in-sequence delivery of downlink frames for Class C devices, the AS should implement a flow control mechanism to send downlink frames one-by-one as follows.
  - Send a downlink frame (FCntDn X)
  - Wait for Downlink Frame Sent Report
    - If SUCCESS: send the next downlink frame (FCntDn X+1).
    - If failed: it is up to AS to decide if a retransmission is required.

(1) The MAC layer of the end-device shall drop downlink frames that are received out of sequence.

7.1 Downlink Frame

7.1.1 URL Base Path

The URL base path for Downlink Frame API v1 is `/thingpark/lrc/rest/downlink`.

The URL base path for Downlink Frame API v2 is `/thingpark/lrc/rest/v2/downlink`.

The only difference between API v1 and API v2 are the HTTP response codes:

- HTTP response codes for the API v1 are described in *7.1.5 HTTP Response Codes with API v1*.
- HTTP response codes for the API v2 are described in *0*

Code	`350 "Token bucket limit reached"`
Description	The downlink packet cannot be sent because the downlink token bucket limit is reached, and the DROP strategy is used.

- HTTP Response Codes with API v2
- .

7.1.2 Payload Encryption

The payload must be provided in non-encrypted form by the AS and will be encrypted by the LRC. The FCntDn is always computed by the LRC.
The following HTTP/POST message format is used to tunnel the radio frame payload and associated metadata from the target Application Server to the LRC. The Application Server acts as an HTTP client and the reverse HTTP proxy (PROXY_HTTP server) acts as an HTTP server. Rerouting of the HTTP request to the primary LRC or the backup LRC is handled by the reverse HTTP proxy.
The LoRaWAN® MAC message integrity code (MIC) is always computed by the LRC, as part of the MAC frame formatting. The MAC payload may either be encrypted by the application or by the LRC (see the table above).
This POST command can be generated easily by tools such as curl or POSTman:

```
curl -H "Content-type:application/x-www-form-urlencoded" -X POST "https://api.thingpark.com/
thingpark/lrc/rest/downlink?DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.com
&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&Token=ea8f31d2299cbece8e180a3012766c4df15fe3cf2e
142d9fdf4035b5894ec886"
```

7.1.3 Query Parameters

The following query parameters are used to generate downlink frames.

Parameter	Description
`DevEUI` (Mandatory)	Target device IEEE EUI64 in hexadecimal format (representing 8 octets) Syntax: STRING (Hexadecimal representation)
`FPort` (Optional)	Target port (in decimal format). This query parameter is needed only if the Application Server provides a payload. Syntax: NUMBER (Unsigned integer: `1..224`)
`Payload` (Optional)	Hexadecimal payload. The hexadecimal payload will be encrypted by the LRC cluster.

Parameter	Description
	The payload can be omitted when `FlushDownlinkQueue=1`: in this case no downlink payload is inserted in the AS downlink queue of the device after the purge. Syntax: STRING (Hexadecimal representation)
`Confirmed` (Optional)	A value of `Confirmed=0` requests transmission of an UNCONFIRMED downlink frame. A value of `Confirmed=1` requests transmission of a CONFIRMED downlink frame. Default value is `Confirmed=0` (UNCONFIRMED). Syntax: NUMBER (Unsigned integer: `0..1`)
`FlushDownlinkQueue` (Optional)	Empties the device AS downlink queue of the device (Boolean). When this parameter is set to `FlushDownlinkQueue=1`, the AS requests the LRC to purge the AS downlink queue of the device. If a payload is provided, it is inserted in the AS downlink queue of the device after the purge. Syntax: NUMBER (Unsigned integer: `0..1`)
`ValidityTime` (Optional)	Associates the AS downlink payload with an expiration date (ISO 8601 timestamp or Duration in seconds) in the device AS downlink queue. If the AS downlink payload has not yet been sent to the device, the AS downlink payload will be discarded by the LRC when the expiration date is reached. Syntax: STRING (ISO date/time) or NUMBER (Unsigned integer)
`AS_ID` (Mandatory)	Application Server ID, as provisioned in the AS Profile. The LRC will check that the Application Server is authorized to send downlink command to the device. Syntax: STRING
`Time` (Mandatory)	ISO 8601 time of the request. **Note** In the URL of the HTTP request, use `%2B` ASCII code for the `+` character and the `%3A` ASCII code for the `:` character. Syntax: STRING (ISO date/time)
`Token` (Mandatory)	Security token to sign the downlink frame. Syntax: STRING (256 bits hexadecimal)
`CorrelationID` (Optional)	64 bits ID used to correlate the downlink frame with the associated downlink frame sent report.

Parameter	Description
	When this parameter is provided, it is sent back in the associated downlink frame sent report. Syntax: STRING (64 bits hexadecimal)

7.1.4 Sample of HTTP Request

Here is a sample of an HTTP request for a downlink frame using API v1.

```
>> POST https://api.thingpark.com/thingpark/lrc/rest/downlink?DevEUI=000000000F1D8693&FPort=
1&Payload=00&AS_ID=app1.sample.com&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&CorrelationID=
4434704901C7450B&Token=ea8f31d2299cbece8e180a3012766c4df15fe3cf2e142d9fdf4035b5894ec886
```

Here is a sample of an HTTP request for a downlink frame using API v2.

```
>> POST https://api.thingpark.com/thingpark/lrc/rest/v2/downlink?DevEUI=000000000F1D8693&FPo
rt=1&Payload=00&AS_ID=app1.sample.com&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&Correlation
ID=4434704901C7450B&Token=ea8f31d2299cbece8e180a3012766c4df15fe3cf2e142d9fdf4035b5894ec886
```

7.1.5 HTTP Response Codes with API v1

The table below gives the description of the HTTP response codes of the LRC.

Code	`200 "Request queued by LRC"`
Description	Request accepted and queued until the class A device opens Rx slots by sending an uplink. In the case of a class C device, the downlink command will be sent as soon as the LRR base station radio is available and the maximum regulatory Tx duty cycle allows transmission.
Code	`350 "Invalid DevEUI"`
Description	The provided DevEUI does not match an existing Device or Multicast Group.
Code	`350 "Confirmed downlink is not authorized for this device"`
Description	The request for transmission of a confirmed downlink packet was rejected by the LRC due to the absence of the **ackedDownlinkFrame** feature flag in the Connectivity plan associated to the device.
Code	`350 "Invalid LoRa port"`
Description	Sending on port non included in `1..224` range is unauthorized from the tunneling interface.
Code	`350 "Security Check. AS_ID is mandatory"`
Description	Speaking to this device needs `AS_ID`. The Application Server authorization has been activated for this device and the application must be identified.
Code	`350 "Security Check. missing timestamp/token"`
Description	`Time` and `Token` query parameters are mandatory when the Application Server authentication is activated.
Code	`350 "Security Check. bad AS_ID"`
Description	`AS_ID` is not declared in the database or is not authorized for the targeted device

Code	`350 "Security Check. Server Decrypt Error"`
Description	Missing or badly formatted security token.
Code	`350 "Security Check. malformed ISO8601 time"`
Description	An ISO 8601 date/time must be used (`YYYY-MM-DDThh:mm:ss.s+|-hh:mm`) representing a local time with a time zone offset in hours and minutes.
Code	`350 "Security Check. Invalid downlink frame timestamp"`
Description	The time deviation between the frame generation by the Application Server and the reception by the LRC exceeds the MAX deviation configured in the AS profile.
Code	`350 "Security Check. bad token"`
Description	Token was not accepted by the LRC.
Code	`350 "ValidityTime expired or invalid"`
Description	The date or duration in the `ValidityTime` attribute is wrongly formatted or is invalid (for instance, date in the past).
Code	`350 "Payload too big or invalid size"`
Description	The payload size is greater than 5000 characters or is not a multiple of 2.
Code	`404 "No Base Station Available"`
Description	The request for transmission of the downlink packet was rejected by the LRC because no base station near to the device is connected.
Code	`350 "Invalid payload size according to currents DR"`
Description	The payload is too big to be transmitted according to the data rate currently used by the device.
Code	`350 "Invalid CorrelationID. Must be a 64 bits hexadecimal value encoded as string"`
Description	The format of the correlation ID is not valid: it must be a 64 bits hexadecimal string.
Code	`350 "Invalid payload. Must not be empty"`
Description	The payload is empty.
Code	`350 "Payload must be provided encrypted with the downlink counter value"`
Description	The payload is provided unencrypted but the LRC is not able to encrypt it (AppSKey is not available).
Code	`350 "Token bucket limit reached"`

Description	The downlink packet cannot be sent because the downlink token bucket limit is reached, and the DROP strategy is used.

7.1.6 HTTP Response Codes with API v2

HTTP response when the downlink frame is accepted:

- HTTP Response status code: **202**
- HTTP Response body: empty.

HTTP response when the downlink frame is rejected:

- HTTP Response status code: **409**
- HTTP Response body: JSON document

```
{
  "code": <code>,
  "message": "<message>"[,
  "xxx:": <additional info 1>][,
  "yyy:": <additional info 2>]
  ...
}
```

  - The code is mandatory (see table below) and is an immutable unsigned integer.
  - The message is mandatory (see table below) and is an ASCII string.
  - On an error basis, additional information elements (xxx, yyy…) may be added to the JSON document (see table below).

The table below gives the description of the error codes.

Error Code	Error Message	Additional info 1	Additional info 2
100	Invalid DevEUI	-	-
101	Downlink counter value already used	Attribute: `expected` Type: Integer Description: Expected FCnt	-
102	Downlink counter value increment too large	Attribute: `expected` Type: Integer Description: Expected FCnt	-
103	Confirmed downlink is not authorized for this device	-	-

Error Code	Error Message	Additional info 1	Additional info 2
104	Invalid LoRa port	-	-
105	Security Check: AS_ID is mandatory	-	-
106	Security Check: Missing timestamp	-	-
107	Security Check: Bad AS_ID	-	-
108	Security Check: Server Decrypt Error	-	-
109	Security Check: Malformed ISO 8601 time	-	-
110	Security Check: Invalid downlink frame timestamp	-	-
111	Security Check: Bad token	-	-
112	Validity time expired or invalid	-	-
113	Payload too big or invalid size	Attribute: `maxSize` Type: Integer Description: When set this means that the max payload size was reached. When not set this means that the payload size is invalid (must be a multiple of 2).	-
114	Downlink transmission disabled	-	-
115	No Base Station Available	-	-
116	Invalid payload size according to currents DR	Attribute: `rx2DR` or `pingSlotDR` Type: Integer Description: MAX Data Rate associated to RX2	Attribute: `maxSize` Type: Integer Description: Max payload size according to RX2

Error Code	Error Message	Additional info 1	Additional info 2
		(class A/C) or PingSlot (class B). The current RX1 DR is not checked.	(class A/C) or PingSlot (class B). The current RX1 DR is not checked.
117	Invalid Correlation ID. Must be a 64 bits hexadecimal value encoded as string	-	-
118	Invalid payload. Must not be empty	-	-
119	Payload must be provided encrypted with the downlink counter value	-	-
121	Token bucket limit reached	-	-

7.1.7 Sample CURL Command

Here is a sample CURL command to send a downlink payload to a device with `DevEUI=000000000F1D8693` using API v1:

```
curl -H "Content-type:application/x-www-form-urlencoded" -X POST "https://api.thingpark.com/
thingpark/lrc/rest/downlink?DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.com
&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&Token=ea8f31d2299cbece8e180a3012766c4df15fe3cf2e
142d9fdf4035b5894ec886"
```

Here is a sample CURL command to send a downlink payload to a device with `DevEUI=000000000F1D8693` using API v2:

```
curl -H "Content-type:application/x-www-form-urlencoded" -X POST "https://api.thingpark.com/
thingpark/lrc/rest/v2/downlink?DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.
com&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&Token=ea8f31d2299cbece8e180a3012766c4df15fe3c
f2e142d9fdf4035b5894ec886"
```

Note

- The ThingPark Enterprise network may queue up to five messages per device. The network uses the **FPending** flag defined in the LoRaWAN® protocol to inform the device that additional messages are queued. Messages will be sent, one at a time, in the receive window following the next uplink message received from the device.

7.2 Downlink Confirmed Application Server Payload

Unconfirmed downlink messages are not acknowledged at LoRaWAN® level. Therefore, the network and the tunnel mode Application Server does not know whether the messages have been received or not.
Confirmed downlink messages are acknowledged by the target device, but the LoRaWAN® Specification -section 4.3.1.2 Message acknowledge bit and acknowledgement procedure *(ACK in FCtrl)*, lets the device free to send delayed ACKs. Therefore, it is not possible to let the network manage retransmissions.
When the LRC receives a possibly empty uplink message (with no payload) with ACK set in the `FCtrl` field, the LRC will add an `ACKbit` flag in the XML metadata of the uplink frame sent to the Application Server. The retransmit policy relies on the Application Server.

7.3 Application Server Authentication of Downlink Frame

7.3.1 Securing Downlink Frame Principles

Securing downlink frame is implemented with the following requirements.
The AS cannot send downlink POST if:

- The AS is not in possession of the Tunnel Interface Authentication Key .
- The AS has not been authorized to send downlink packet to the device.

The AS ID and Tunnel Interface Authentication Key are part of the Generic Application configuration.
For more information about the Tunnel Interface Authentication Key, see *4 Connectivity*.

7.3.2 Token Computation Details

The Token must be computed by the Application Server as follows.

1. The downlink message *<query-parameters>* (Query parameters must include the `AS_ID` and the `Time` query parameters) are constructed WITHOUT the Token:
   An example of *<query parameters>* is:
   `DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.com&Time=2016-01-11T14:28:00.333+02:00`

   where: `2016-01-11T14:28:00.333+02:00`
   contains the `+` character in the time zone designator of the `Time` parameter.

2. The *<token>* is computed as SHA-256(<query-parameters><LrcAskey>)
   An example of *<token>* is:
   `SHA-256(DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.com&Time=2016-01-11T14:28:00.333+02:0046ab678cd45df4a4e4b375eacd096acc)`

   where: `46ab678cd45df4a4e4b375eacd096acc`
   is the 128 bits pre-shared Tunnel Interface Authentication Key (lower case hexadecimal

string representation) between the Application Server and the LRC as defined in the Generic Application configuration.

where: `2016-01-11T14:28:00.333+02:00`
contains the `+` character in the time zone designator of the `Time` parameter.

3. The *<token>* is encoded as a hexadecimal string (An example of *<encoded-token>* is: `63a4ec6532937c9bcba109a75f731d6dc192c9df662dee56757634a8a6dc3f4c`) AND added at the end of the query parameters line of the HTTP request.
   An example of *<encoded-token>* is:
   `https://api.thingpark.com/thingpark/lrc/rest/downlink?DevEUI=000000000F1D8693&FPort=1&Payload=00&AS_ID=app1.sample.com&Time=2016-01-11T14%3A28%3A00.333%2B02%3A00&Token=63a4ec6532937c9bcba109a75f731d6dc192c9df662dee56757634a8a6dc3f4c`

   where: `2016-01-11T14%3A28%3A00.333%2B02%3A00`
   contains the `%2B` ASCII code for the `+` character and the `%3A` ASCII code for the `:` character.

8 REFERENCE INFORMATION

8.1 What's New

8.1.1 What's New in Release 6.1

- Uplink frame report (LRC to AS) has been enhanced:
  - The <Lrrs> field reports up to 10 LRRs which reported the packet before the expiration of the 250 ms deduplication window.
  - The <DevLocTime>, <DevLAT>, <DevLON>, <DevAlt>, <DevLocRadius>, <DevAltRadius>, <DevNorthVelocity> and <DevEastVelocity> fields report the last known geolocation.
  - The <NwGeolocAlgo> and <NwGeolocAlgoUsed> fields provide information about the algorithm used to compute the network geolocation.
  - The <DriverCfg> field reports the metadata for driver selection.
- Location report (LRC to AS) has been added:
  - The <DevEUI_location> report gives the current geolocation data of the device as soon as available from the location server.
- Downlink frame (AS to LRC) has been enhanced:
  - An API v2 implementing new HTTP response codes is available.
  - `Payload` and `FPort` query parameters are optional when the `FlushDownlinkQueue` query parameter is set to 1: in this case no downlink payload is inserted in the AS downlink queue of the device after the purge.
  - A new Downlink Frame HTTP response code is returned when the downlink token bucket limit is reached, and the DROP strategy is used:
    - API v1: `350 "Token bucket limit reached"`
    - API v2: `409 {"code":121,"message":"Token bucket limit reached"}`

8.1.2 What's New in Release 6.0

- Uplink frame report (LRC to AS) has been enhanced:
  - The <DynamicClass> field reports the LoRaWAN® class currently used by the Device.
  - The <ClassBPeriodicity> field reports the Class B periodicity in seconds requested by the device.
  - The <Frequency> field reports the frequency in MHz of the radio channel used to receive the frame.
- Downlink frame sent report (LRC to AS) has been enhanced:
  - The <TransmissionSlot> field reports the slot used for downlink frame transmission.
  - The <DeliveryFailedCause3> field provides information about the downlink delivery status on ping slot for Class B devices.
  - The <CorrelationID> field is an identifier used to correlate the downlink frame sent report with the unicast downlink frame previously submitted.

- The <Frequency> field reports the frequency in MHz of the radio channel used to send the frame.

- Downlink frame (AS to LRC) has been enhanced:
  - The CorrelationID query parameter is an identifier used to correlate the downlink frame with the associated downlink frame sent report.
- Added new Downlink Frame HTTP response code: `350 "Invalid CorrelationID. Must be a 64 bits hexadecimal value encoded as string".`
- Updated Downlink Frame HTTP response code: `350 "Invalid LoRa port".`
- Added new Notification report.

8.1.3 What's New in Release 5.2.2

- New "A5" delivery failed cause in downlink frame sent report for radio board error.
- Renamed LRC-AS Key as Tunnel Interface Authentication Key.

8.1.4 What's New in Release 5.1

- Uplink frame report (LRC to AS) has been enhanced:
  - The <ConfAFCntDn> field reports the applicative downlink counter CONFIRMED by the device when LoRaWAN® 1.1 is used.
  - The <AFCntDn> field reports the applicative downlink counter to be used for the next downlink frame when LoRaWAN® 1.1 is used.
  - The <TxPower> field reports the transmission conductive power or EIRP of the device.
  - The <NbTrans> field reports the number of transmissions for each uplink message computed by the LRC.
- Downlink frame sent report (LRC to AS) has been enhanced:
  - The <AFCntDn> field reports the applicative downlink counter for this packet when LoRaWAN® 1.1 is used.
- Downlink frame (AS to LRC) has been enhanced:
  - The AFCntDn query parameter provides the applicative downlink counter to be used for the downlink frame when the Application Server encrypts the payload and LoRaWAN® 1.1 is used.

8.1.5 What's New in Release 5.0

- The <Lrrid> XML element in the uplink frames now reports the ID of the LRR that received the packet with the best SNR.
- The <Lrrs> XML element in the uplink frames now reports 3 LRRs which received the packet with the best SNR.
- New "DC" delivery failed cause in downlink frame sent report for No GPS-synchronized LRR detected by the LRC.
- New "DD" delivery failed cause in downlink frame sent report for No LRR connected detected by the LRC.
- New "DE" delivery failed cause in downlink frame sent report for DC not allowed by the peering operator.

- New Downlink Frame HTTP response codes.

8.1.6 What's New in Release 4.3

- New "DB" delivery failed cause in downlink frame sent report for max downlink dwell time check. Max downlink dwell time is defined per RF Region in the LoRaWAN® specification.

ABOUT ACTILITY

Actility is an industry leader in LPWAN (Low Power Wide Area) large scale infrastructure with ThingPark™, the new generation standard-based M2M communication platform. Actility's ThingPark Wireless™ network provides long-range coverage for low-power sensors used in SmartCity, SmartBuilding and SmartFactory applications. Actility also provides the ThingPark X which provides big data storage for sensor data and exposes sensor function through an open API allowing developers to provide vertical applications on top of rolled out sensors. To help vendors transform their sensors, Actility provides the ThingPark IoT platform which include embedded software solutions and cloud solutions to help devices connect to innovative applications. Via the ThingPark Market, an online marketplace engine dedicated to the IoT sensors, applications and network solutions, Actility enables the roll-out of new innovative IoT services for sensor vendors and network solution vendors. Actility is a founding member of the LoRa Alliance®: the largest, most powerful standards-based effort to enable the Internet of Things (IoT). Visit www.actility.com.